**Build a Build Script**

\***Note**: this project is provided by ***Codecademy.com***

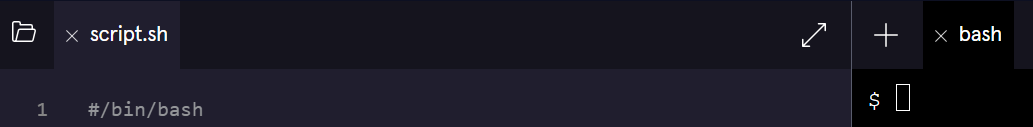
Below is the set up provided by Codecademy.com.

A screenshot of a computer

Description automatically generated

In this project, I created a release script to copy certain files from a **source** directory into a **build** directory.

1. I first looked at the **build** and **source** folders. The objective of the script is to copy files from **source** to **build**, with a couple of exceptions and modifications. I started on the script by adding a header to **script.sh**, identifying the type of script.



1. I created an echo to welcome the user and tested out the script in the terminal using **./script.sh**.

A black and blue background

Description automatically generated with medium confidence

1. Since this is a new build, I verified that the **changelog.md** is the current release version. The first line (highlighted in yellow) of the file contains a version number with markdown formatting.

A screenshot of a computer

Description automatically generated

I read the first line of this file into a variable **firstline.**

* **Note:** Using the “**head**” argument returns the first 10 lines. The argument “**-n**” is used to specify the number of lines. I added **1** after the -n argument to specify only the first line. Shown in line 3:

**A screen shot of a computer

Description automatically generated**

1. I wanted just the version number without the markdown formatting. The command ***read*** can be used to split a string into an array using the ***-a*** argument.
   1. I split the string **firstline** into the array **splitfirstline.**
   2. The syntax for splitting a string **foo** into an array **bar** is:

A screenshot of a computer

Description automatically generated

* + 1. **Note:**  Do not type this in.
  1. Here’s is the result of what I typed in line 4 of the script:

A screen shot of a computer

Description automatically generated

1. I set the value of the version of the script. It is located in **index 1** of the array **splitfirstline**.

A red line on a blue background

Description automatically generated

* 1. The syntax for accessing the value of **index** of an array **foo** is:

A white text on a black background

Description automatically generated

* + 1. **Note**: Do not type this in.
  1. I saved the version to a variable **version**. Shown in line 5:

A screenshot of a computer

Description automatically generated

* 1. I printed a statement to the terminal (e.g. “[statement]” $version) notifying the user of the version they are building. Shown in line 6:

A screenshot of a computer screen

Description automatically generated

* 1. Result of outputting script.sh in the bash terminal:

Screens screenshot of a computer screen

Description automatically generated

1. I printed a statement asking the user if they wanted to exit the script if they needed to make changes to the version. I assigned the response to the variable **versioncontinue**. This is what I typed in lines 7, 8 and the output in the bash terminal:

A screenshot of a computer

Description automatically generated

1. I added a conditional. If the user said “1” to the continue question, the rest of the script was executed, and the response was “OK”. If the user did not, the respond was “Please come back when you are ready”.
   1. **Note**: you can choose your own responses.
   2. This is what I typed from lines 10 to 15 and the output in bash:

A close up of a screen

Description automatically generated

1. I copied every file from **source** to **build**. Within the positive condition (Where I told the user “OK”), I started by iterating over all the files in the **source** directory and printed their names to the terminal.
   1. This is what I typed from lines 13 through 16 and the output in the bash terminal.

A screen shot of a computer screen

Description automatically generated

1. I decided I don’t want to copy the file name **secretinfo.md** to the build source. To do this, I created an if/else statement within the **do** loop that informs the user what was copied and what was not.
   1. **Note**: the **cp “[**pick file name**]” [directory name]/.** command will copy a file to a directory.
      1. In my case, I used the **cp** **$filename build/.** command in the script (on line 21 in the script
   2. This is what I type from lines 16 to 22 and the output in the bash terminal.

A screenshot of a computer screen

Description automatically generated

* 1. I used the **ls build/** command in the bash terminal to check if the files were copies.

A screenshot of a computer screen

Description automatically generated

1. I then navigated to the build directory in the bash terminal to show the results.
   1. To show the results in the bash terminal, I used the **ls** command to show the root directory. Then I navigated to the build directory using the **cd build** command. Finally, I use the **ls** command to show the files copied to the build directory. Finally, I navigated back to the root directory I have been working in using the **cd ../** command.

A screen shot of a computer code

Description automatically generated

1. This navigation can also be completed in the script. Which is shown in lines 28 through 31 (which includes a print command referencing on what’s contained with version 11.2.3 of the build directory) and the script output in the bash terminal.

A screenshot of a computer program

Description automatically generated

**END OF PROJECT**